Internship Report

Intern: Alireza Habibzadeh
Supervisor: Dr. Marco Scigliuzzo
Reporting to: Prof. Tobias J. Kippenberg

Summer 2023

École Polytechnique Fédérale de Lausanne (EPFL)
Laboratory of Photonic Integrated Circuits and Quantum Measurements (LPQM)

Introduction

During my summer internship, I worked on a main project and two other tasks. As my main project, I worked on creating an Automatic Qubit Calibrator using Quantum Machines QUA API. As my other tasks, I developed a Python driver and a web-based graphical user interface to control a physical switch inside the Bluefors Dilution Refrigerator Measurement Systems in the lab. The second task involved setting up and automating the MPI TS2000-D Probe Station and the Keithley 4200A-SCS Parameter Analyzer to make them communicate for automated measurements. This report details the methods, challenges, and insights gained from each project, as well as my learning experiences during the internship.

1 Automatic Qubit Calibrator

Introduction

We developed an automatic calibrator system for Quantum Machines® controllers (OPX+ and Octave). Similar to Google's approach[1] we implemented eight *"calibration nodes"* to employ spectroscopy techniques to measure and analyze reflection data (S_{11}) from the superconducting qubits. We have also implemented a database and API for communication among these nodes. Automating the calibration process, formerly done with Vector Network Analyzers, lets us streamline measurements. This allows researchers to analyze temporal shifts by continuously monitoring resonator frequencies, Qubit frequencies, T_1, and more.

1.1 Qubit Characterization

Qubit characterization is an important process in quantum computing, particularly right after the fabrication of each quantum bit. The characterization involves determining parameters like resonator frequency, qubit frequency, optimal π-pulse length for qubit excitation, and the relaxation time T_1. Traditionally, this process has been manual, demanding considerable time and energy for each fabricated sample.

The goal of this project is to automate the process of qubit characterization. Automation is not only a matter of convenience but a necessity due to the labor-intensive nature of the manual process. With every new setup, change in the setup, or even temporal variations in qubits, manual recalibration is required. This is a time-consuming task, often involving complex equipment like Vector Network Analyzers (VNAs).

In my project, I first comprehended and replicated the entire manual process using Quantum Machines. Unlike traditional setups, Quantum Machines, comprising a controller and a mixer, are designed for automation and coding. This feature makes them ideal for the purpose of automating the characterization process. The aim was to make this process as automatic and user-friendly as possible.

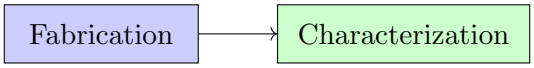


Figure 1: Qubit characterization provides feedback for the fabrication. (Both take time and energy!)

1.2 Dispersive Readout

Dispersive readout is a fundamental technique in quantum computing for qubit state measurement. This method relies on the interaction between a qubit and a resonator, which can be described by the Jaynes–Cummings Hamiltonian:

$$H_{JC} = \omega_r \left(a^\dagger a + \frac{1}{2} \right) + \frac{\omega_q}{2} \sigma_z + g(\sigma_+ a + \sigma_- a^\dagger) \tag{1}$$

where ω_r is the resonator frequency, ω_q is the qubit frequency, g is the coupling strength, $a^\dagger$ and a are the creation and annihilation operators for the resonator, and σ_+ and σ_- are the raising and lowering operators for the qubit.

In the dispersive regime, the qubit is far detuned from the resonator, compared with their coupling rate. Mathematically, this condition is expressed as:

$$\Delta = |\omega_q - \omega_r| \gg g \tag{2}$$

In this limit, the effective Hamiltonian under the dispersive approximation becomes:

$$H_{disp} = (\omega_r + \chi \sigma_z) \left(a^\dagger a + \frac{1}{2} \right) + \frac{\tilde{\omega}_q}{2} \sigma_z \tag{3}$$

where

$$\chi = \frac{g^2}{\Delta} \tag{4}$$

and

$$\tilde{\omega}_q = \omega_q + \frac{g^2}{\Delta} \tag{5}$$

This results in a shift of the qubit and resonator frequencies, effectively *pushing* each other's frequencies, a phenomenon used in the dispersive readout process.

1.3 Measurement Technique

Measuring the qubit's state in quantum computing is a complex task, particularly because direct access to the qubit is not feasible. Instead, our technique relies on indirect measurement through a resonator coupled to the qubit.

Dispersive Regime and Signal Analysis

In the dispersive regime, the resonator's frequency is significantly different from that of the qubit, allowing for distinct interactions without energy exchange. We send signals to the resonator and analyze the reflected signals to study the qubit. This method is effective because the resonator's behavior changes depending on the state of the qubit, thus providing indirect information about the qubit's state.

Qubit Drive Setup

The qubit drive setup involves a microwave source that supplies a high-frequency signal (xLO). An arbitrary waveform generator (AWG) provides a pulse envelope (s(t)), which may include a low-frequency component, xAWG, generated by the AWG. The IQ-mixer combines these signals to create a shaped waveform $V_d(t)$ with a frequency $x_d = x_{LO} \pm x_{AWG}$, typically resonant with the qubit. The waveform is then translated into a gate sequence, represented by the I and Q components.

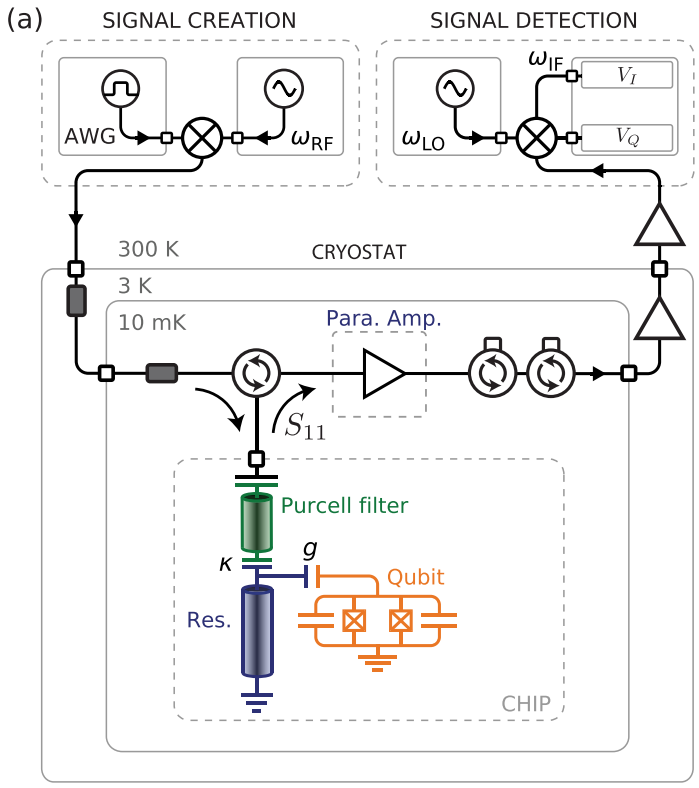


Figure 2: Simplified schematic of the setup used for dispersive qubit readout.[2]

Dispersive Qubit Readout

For dispersive qubit readout, the resonator probe tone is generated and shaped by an AWG. This signal is sent into the cryostat, and the reflected signal S_{11} is amplified and analyzed. The reflected magnitude $|S_{11}|$ and phase response of the resonator vary depending on whether the qubit is in its ground state $|0\rangle$ or excited state $|1\rangle$. In the complex plane representation, each state corresponds to a point composed of the in-plane $\Re(S_{11})$ and quadrature $\Im(S_{11})$ components, allowing for efficient state discrimination.

1.4 The General Problem of Calibration

Each node in a calibration process evaluates certain parameters, but these actions depend on prior calibrations. This introduces a complex network of dependencies, effectively creating a Calibration Graph.

Calibration Graph and Dependency Management

A Calibration Graph is a conceptual representation of these dependencies. This graph must be structured as a Directed Acyclic Graph (DAG), which means it cannot have cycles. If cycles are present, it leads to a dependency loop, making calibration impossible to complete.

In this example DAG, each node represents a calibration step. The arrows indicate the direction of dependency, with calibrated nodes influencing subsequent nodes.

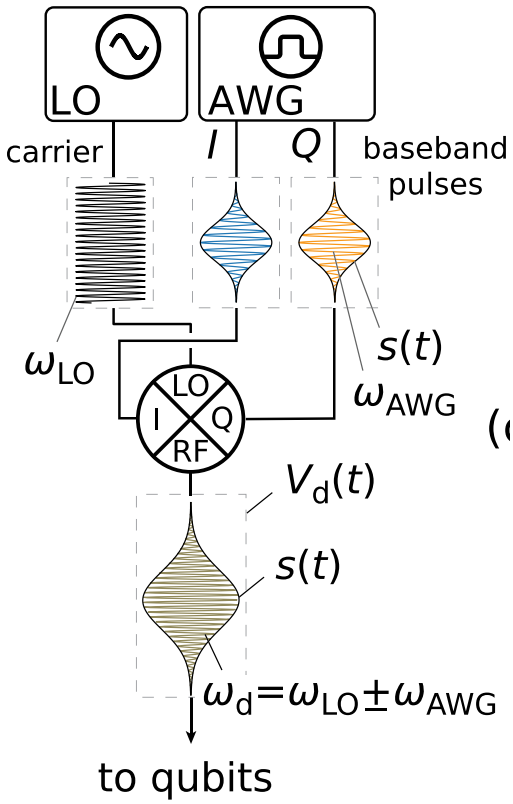


Figure 3: Schematic of a typical qubit drive setup.[2]

Optimus: A Dynamic Calibration System[1]

A noteworthy example of an advanced calibration system is Optimus, proposed by Google. Optimus is designed to keep systems perpetually calibrated, reevaluating and adjusting necessary parameters as they shift over time. This dynamic approach guarantees consistent accuracy in rapidly changing environments.[1]

1.5 Databases

We use two primary databases: the Measurement Database and the Parameter Database.

The Measurement Database

This database is for measurement records and the parameters that were in effect at the time of each measurement. It was previously implemented by Evgenii Guzovskii. The data stored in this database includes:

- Measurement records.
- Parameters at the time of each measurement.

Parameter Database

The Parameter Database is structured to store a variety of information, which includes:

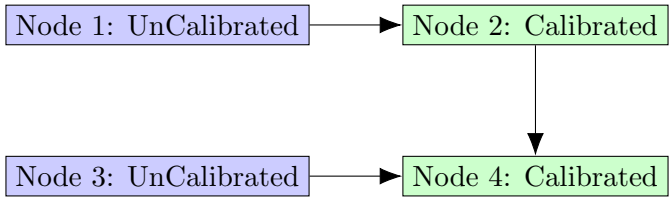


Figure 4: Example of a Directed Acyclic Graph (DAG) in Calibration

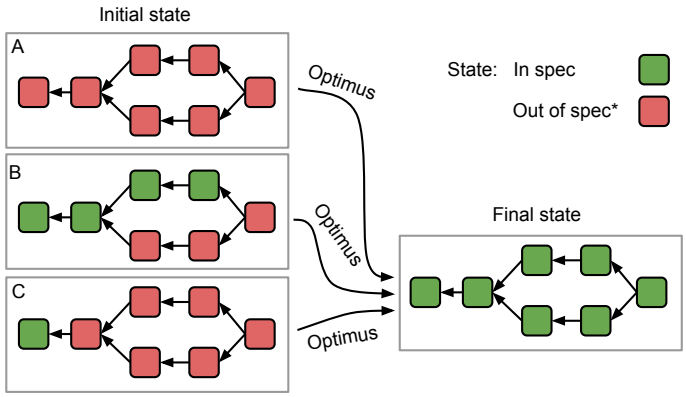


Figure 5: Overview of the Optimus Calibration System

Internally used parameters in this database are critical for calibration processes and include:

- Resonator frequency range.
- Run IDs for measurements linked to the Measurement Database.
- Other relevant calibration parameters.

Parameter Database vs. OneNote

Initially, OneNote was used to track these parameters. However, this method has several limitations:

- Lack of automation capabilities: OneNote does not support automated data entry or updates.
- Temporal analysis difficulty: It's challenging to track and visualize parameter shifts over time in OneNote.
- Inefficiency in handling multiple qubits and setups: OneNote's structure makes it cumbersome to manage data across different qubits and experimental setups.

1.6 Calibration Node

A calibration node is a component in the calibration process. Its functionality can typically be broken down into three main steps:

1. **Run Scan**: Initiating a scan to gather the necessary measurement data.

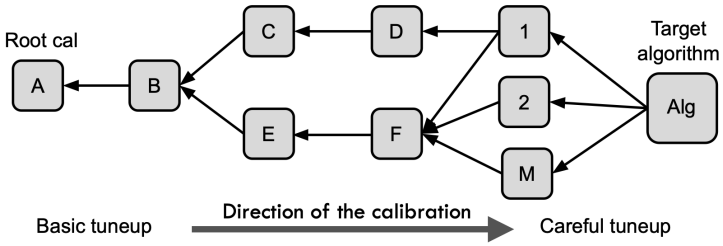


Figure 6: Workflow in the Optimus Calibration System

Field	Description
ID	Unique identifier for each record
Timestamp	Time when the parameter was recorded
Creator	Calibration node who recorded the parameter
Resonator AF	Resonator Frequency
Resonator Working Power	Power at which the resonator operates efficiently

Table 1: Fields in the Parameter Database

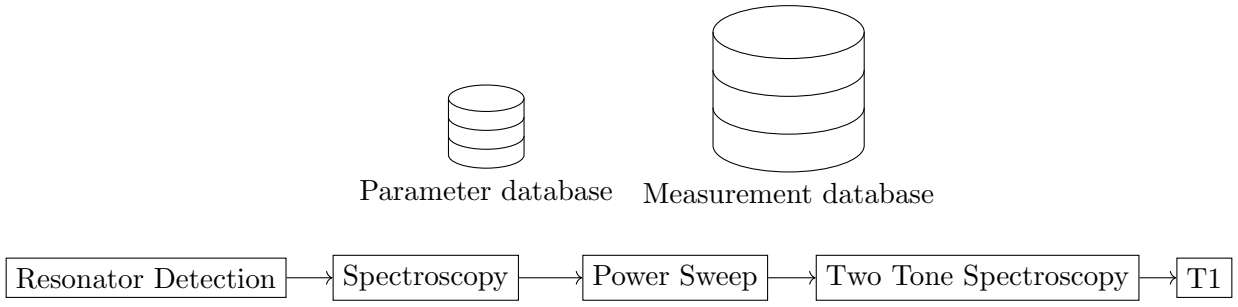
2. **Analyze Data**: Processing the collected data to determine optimal parameter values.
3. **Update Parameters**: Adjusting the system parameters based on the analysis.

Separation of Measurement and Analysis

A key feature of a calibration node is the separation of its measurement and analysis components. This division provides several advantages:

- **Flexibility in Data Usage**: The ability to run the calibrator on previously measured data allows for greater flexibility. This means one can revisit and reanalyze past measurements without rerunning the entire calibration process.
- **Testing with Simulated Data**: This separation could enable the calibrator to run on simulated data. This feature helps for testing and validation purposes, and to check the accuracy of the calibration process before applying it to actual data.

1.7 Our Calibration Graph



1.7.1 Resonator Spectroscopy

This process involves shining light on the resonator to study its properties. The primary goal is to determine various characteristics of the resonator. This is necessary for further qubit operation.

Database Update Post-analysis, the following parameters are updated in the database:

- Resonator frequency
- Resonator frequency range
- Fitting parameters such as internal loss (κ_{int}), external loss (κ_{ext}), and others

Mathematical Expression The transmission coefficient $S_{21}(f)$ of the resonator is given by the equation:

$$S_{21}(f) = 1 - \frac{QQ_e^{-1}}{1 + 2jQ\left(\frac{f-f_0}{f_0}\right)} \tag{6}$$

where Q is the quality factor of the resonator, Q_e is the external quality factor, f is the frequency, and f_0 is the resonator frequency.

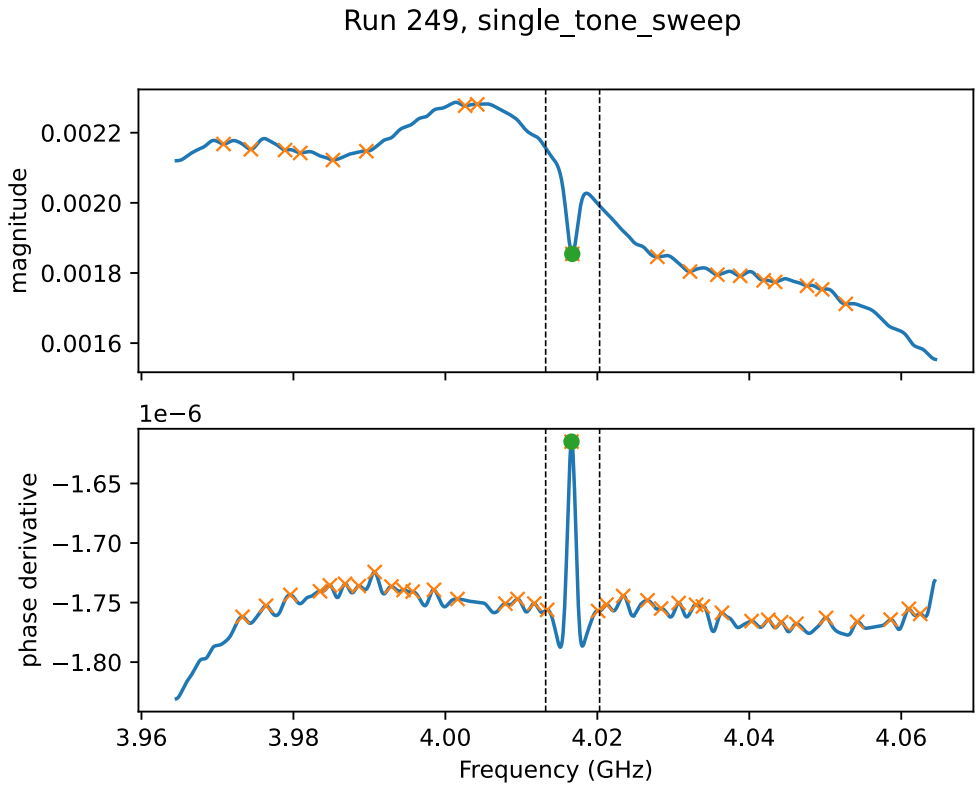


Figure 7: Wide frequency scan to find the resonator.

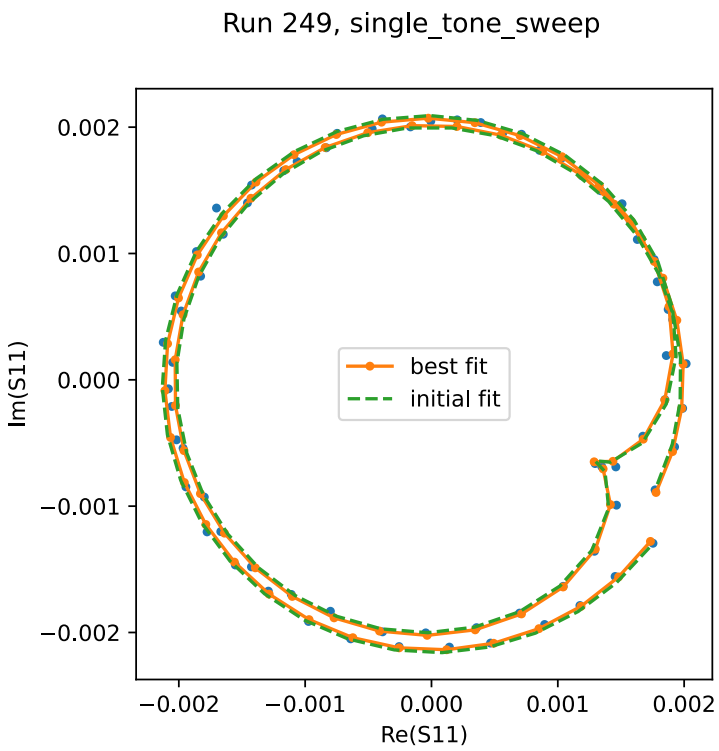


Figure 8: Fitting the resonator.

1.7.2 Sweeping the Readout Power (Punchout)

This node in the calibration graph involves sweeping the signal power and tracking the resonator frequency in the reflected signal.

Goals The objectives of this calibration step are:

- To verify the presence and state of the qubit: By observing how the qubit shifts the resonator's frequency, we can determine whether the qubit is 'alive' or not.
- To identify the optimal operating power for the resonator: Determining the right power level to interact effectively with the resonator.
- To estimate the qubit's frequency: Based on the degree of shift in the resonator's frequency, we can infer the qubit's frequency.

Mathematical Expression The shift in the resonator frequency caused by the qubit is given by the formula:

$$\delta = \frac{g^2}{\Delta} \tag{7}$$

where g is the coupling strength between the qubit and the resonator, and Δ is the detuning between their frequencies.

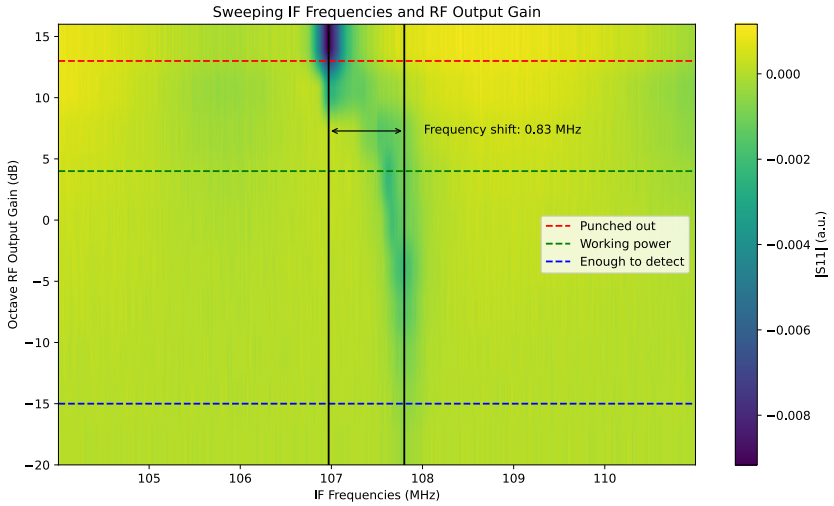


Figure 9: 2D Heatmap of S11 magnitude on a plot of frequencies vs RF output gain

1.7.3 Two-Tone Spectroscopy

Two-tone spectroscopy is another node in our calibration graph. It is primarily used to determine the qubit's resonant frequency, ω_q.

Procedure The process involves:

- Driving the qubit through the resonator at different frequencies.
- Sending a frequency sweep near the estimated qubit frequency through the resonator.
- Measuring the response on the readout line.

The objective is to pinpoint the qubit's resonant frequency by analyzing the resonator's response to these varied frequencies.

Data Analysis The collected data is then plotted to observe the amplitude response. The peak in this plot corresponds to the qubit's resonant frequency, which is required for any further calibration and operation.

1.7.4 PCA Decomposition

In the PCA Decomposition node, we employed *scikit-learn*, a machine learning library for Python, to perform a PCA (Principal Component Analysis) decomposition on our data. This approach significantly improved our signal-to-noise ratio (SNR).

Mathematical Formulation The PCA aims to maximize the sum of squared linear projections. The mathematical formulation is given as follows:

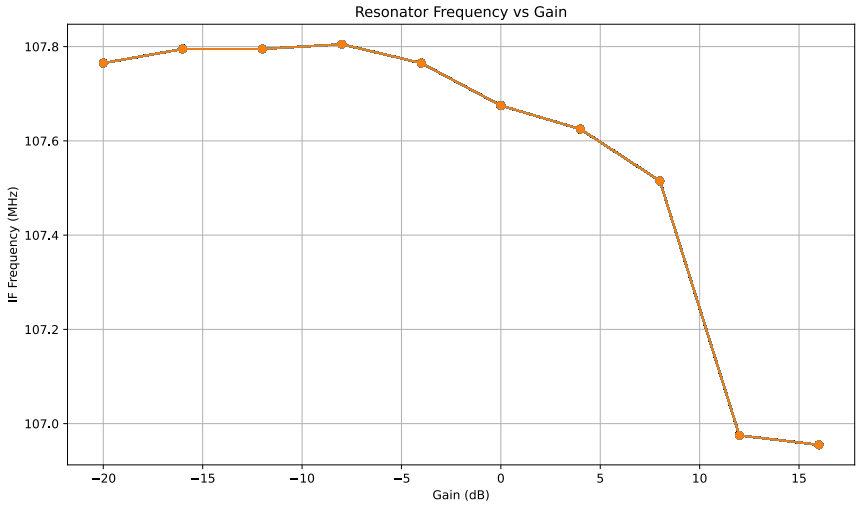


Figure 10: Resonator frequency vs power gain in dB

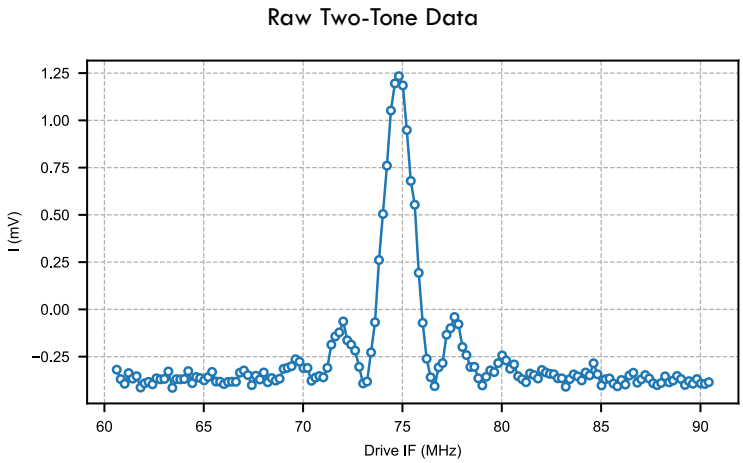


Figure 11: Raw Two-Tone Spectroscopy Data: I vs Drive Frequency

$$\begin{aligned}\text{Sum of squared linear projections} &= \sum_{i=1}^{N}(\vec{x}_i \cdot \vec{v})^2 = \|X\vec{v}\|^2 \\ &= (X\vec{v})^\top (X\vec{v}) \\ &= \vec{v}^\top X^\top X \vec{v} \\ &= \vec{v}^\top (X^\top X)\vec{v} \\ &= \vec{v}^\top C \vec{v},\end{aligned}$$

where C is the covariance matrix of the data. The Lagrange function for this optimization problem is:

$$L = \vec{v}^\top C \vec{v} + \lambda(\vec{v}^\top \vec{v} - 1) \tag{8}$$

Setting the derivative of L with respect to $\vec{v}$ to zero, we get:

$$\frac{\partial}{\partial \vec{v}} L = 2C\vec{v} - 2\lambda\vec{v} = 0 \quad \Longrightarrow \quad C\vec{v} = \lambda\vec{v} \tag{9}$$

This implies that $\vec{v}$ must be one of the eigenvectors of C, and the eigenvector associated with the largest eigenvalue maximizes the sum.

Data Analysis Instead of discarding parts of the data, PCA uses all available data most effectively. We can better separate the signal from the noise by projecting the data onto the principal components.

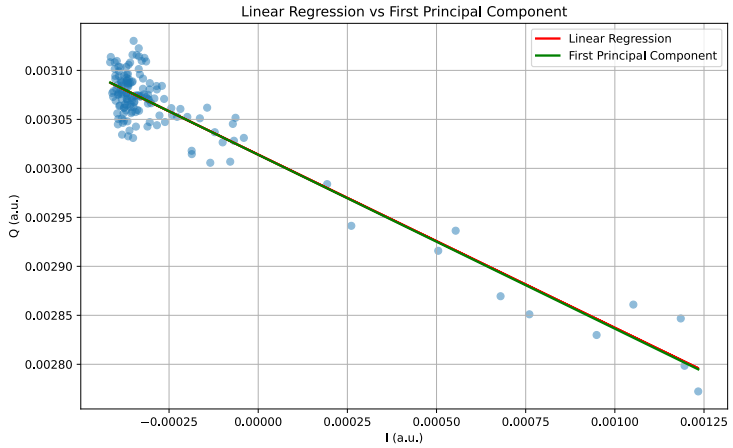


Figure 12: Unrotated Data

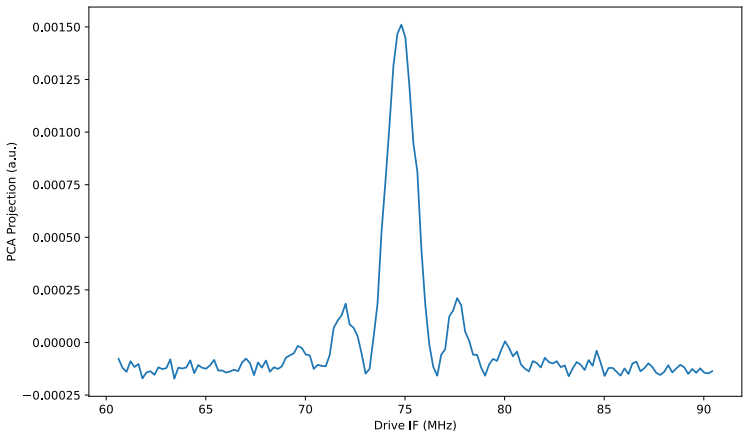


Figure 13: Result in the First Principal Component

1.7.5 Rabi Measurement

The Rabi Measurement node determines the X_π pulse length, which we need in qubit control. The X_π pulse is a specific pulse duration that rotates the state $|0\rangle$ to $|1\rangle$, effectively performing a 180°

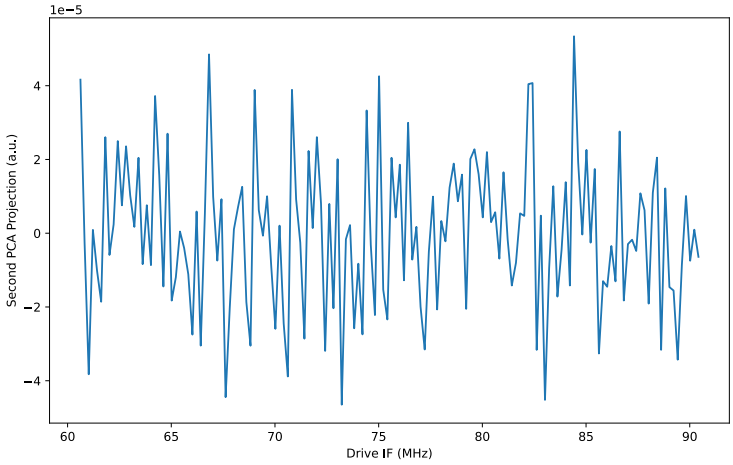


Figure 14: Second Principal Component (Mostly Noise)

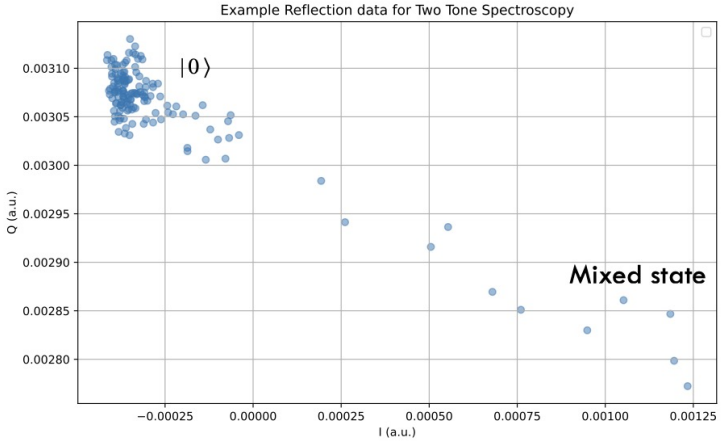


Figure 15: Reflection Data for Two-Tone in IQ Plane

rotation.

Procedure The process for determining the X_π pulse length includes:

- Driving the qubit with a pulse at its resonant frequency ω_q.
- Reading out the qubit state through the resonator to determine the probabilities of being in the state $|0\rangle$ or $|1\rangle$.
- Repeating the experiment for a range of pulse lengths.

Data Analysis The collected data is plotted with the averages of the qubit states against the pulse lengths. The peak in this plot represents the X_π pulse length, indicating the optimal duration for rotating the qubit state from $|0\rangle$ to $|1\rangle$.

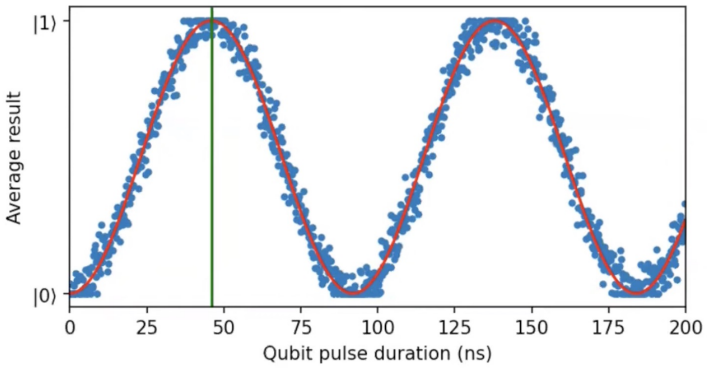


Figure 16: Average Result State vs Qubit Pulse Duration

1.7.6 T1 Measurement

The T1 Measurement node is the final step in our calibration graph, focusing on measuring the energy relaxation time, denoted as T_1, of the qubit.

Procedure The T1 measurement process includes:

- Performing a π-pulse to prepare the qubit in a mixed state, predominantly $|1\rangle$.
- Waiting for a time duration τ before reading out the qubit state.
- Repeating the experiment multiple times for each value of τ to accurately determine the qubit's decay behavior over time.
- Experimenting with a range of delay times τ.

Data Analysis The collected data is plotted against the delay times, and an exponential decay fit is applied to determine the T_1 time constant. This constant represents the characteristic time over which an excited qubit decays to the ground state $|0\rangle$.

Database Update After determining the T_1 value, it is updated in the database.

1.8 Future Plans

Calibrate the Calibrator!

There are tens of parameters in the calibrator which are set manually; like how wide the range of scans for the resonator should be. The calibrator will work if the setup is slightly modified. However, changing the frequency regime in which we work will require manual adjustments to the calibration system. We hope to implement tests or procedures to calibrate these parameters in the calibrator.

Self Diagnosis

The calibrator should detect if the qubit is alive, and halt the process if it finds something wrong with the setup.

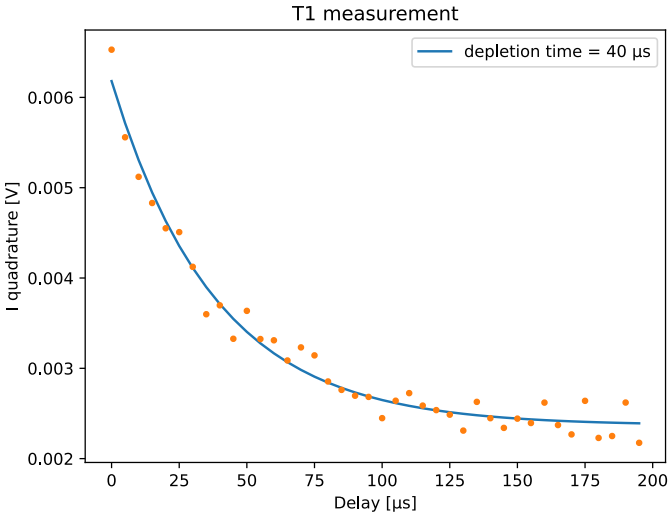


Figure 17: I vs Delay in T1 Measurement

1.8.1 Machine Learning Resonator Detection

As we look ahead, we find several exciting directions for the continuation and enhancement of our project. One notable area is the application of machine learning techniques to improve the efficiency and accuracy of our processes.

A promising avenue for future development is to use machine learning for resonator detection. This approach can potentially replace the classical signal processing to identify and characterize resonators within our system.

Complex Data Utilization Traditionally, resonator detection relies on analyzing phase and amplitude data. However, machine learning offers the ability to use the full complex data, namely the I (In-phase) and Q (Quadrature) components. This comprehensive data usage can lead to more accurate detection.

Methodology The process involves training a deep learning model on a dataset of known resonator characteristics, encompassing a wide range of I and Q data. This model can then learn to identify patterns and signatures of resonators, even in complex or noisy environments.

Advantages By using machine learning:

- We can potentially automate the detection and characterization process, reducing the time and manual effort required.
- The system can become more robust to variations and noise, as deep learning models can learn to filter out irrelevant data while focusing on key features.

Multi-qubit calibrations

We have implemented the calibrator with one qubit in mind. Of course, characterizing multiple standalone qubits which are not coupled is also possible with the current system, characterizing

multiple coupled qubits is what needs to be implemented in the future.

2 The Switch Controller

2.1 Task Description

In this project, I focused on automating the control of Radiall Switches used in the lab's Bluefors fridges to switch between different samples. The main issues with the existing manual system included incorrect mapping, excessive heating of the fridge due to pulse lengths, inconvenience, and the potential for human error.

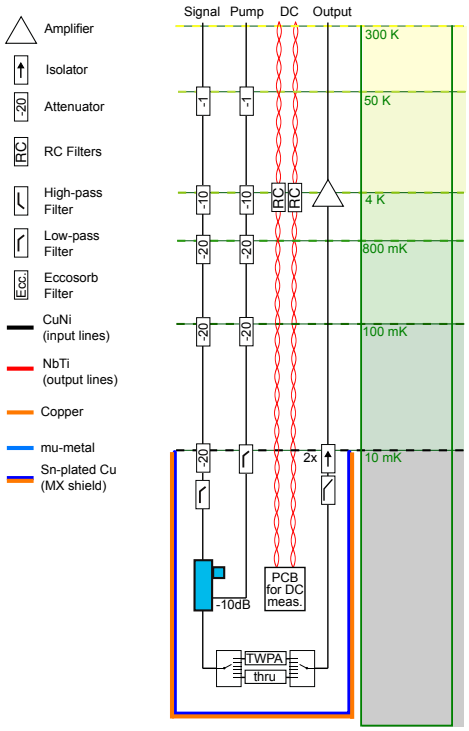


Figure 18: Schematic of the fridge showing the placement of the Radiall Switches

2.2 Implementation

I developed a Python package to communicate with the network switches, a task necessitated by the lack of existing drivers for commercial switches. The package, available on GitHub at `https://github.com/alirezahabib/dscript-driver`, allows for precise control of the switching mechanism. The code example below demonstrates the basic usage of the package:

Input

```
import dscript

dscript.ip_address = '169.254.100.1'

```

```
dscript.reset_relays()
dscript.get_relays()
dscript.toggle_relay(2)
dscript.get_relays()
```

Output

```
INFO:root:Resetting all relays to False
INFO:root:Fetching data from 169.254.100.1
INFO:root:Toggling relay 2
INFO:root:Request successful
INFO:root:Fetching data from 169.254.100.1
[False, False, False, False, False, False, False, False, False, False, False,
    False, False, False, False, False, False, False, False, False, False, False,
    False, False, False, False, False, False, False, False, False, False]
INFO:root:Toggling relay 2
[False, True, False, False, False, False, False, False, False, False, False, False
    , False, False, False, False, False, False, False, False, False, False, False,
    False, False, False, False, False, False, False, False, False, False]
```

Following this, I created a user-friendly graphical user interface (GUI) to enhance usability.

2.3 Results and Discussion

The implementation significantly improved the switching process. Measurements using an oscilloscope verified that the pulses are now shorter, leading to less heating of the fridge. This was further evidenced by a lower temperature increase recorded in the fridge with each switch. The GUI provided a more convenient and error-free way to control the switches.

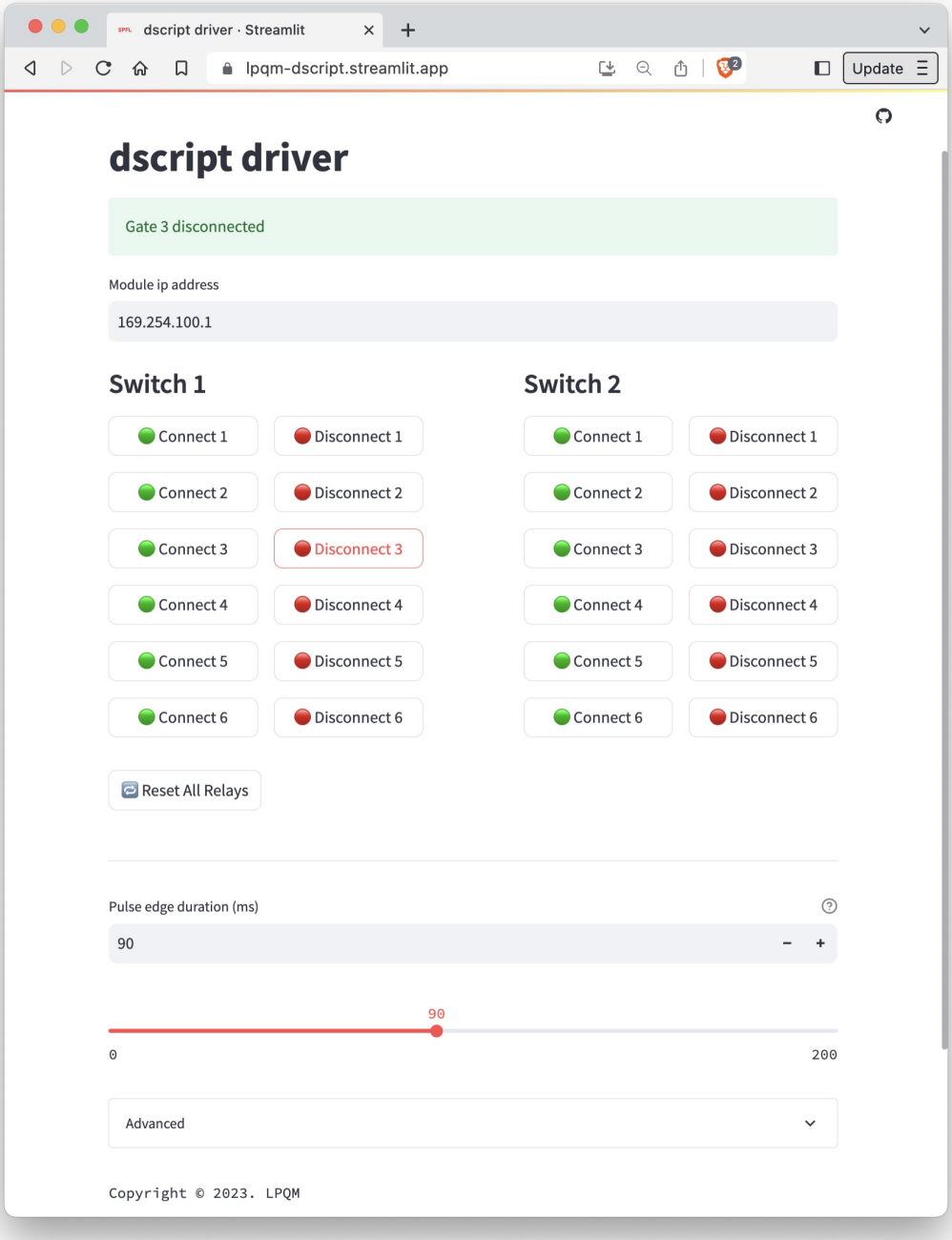


Figure 19: Screenshot of the GUI developed for switch control

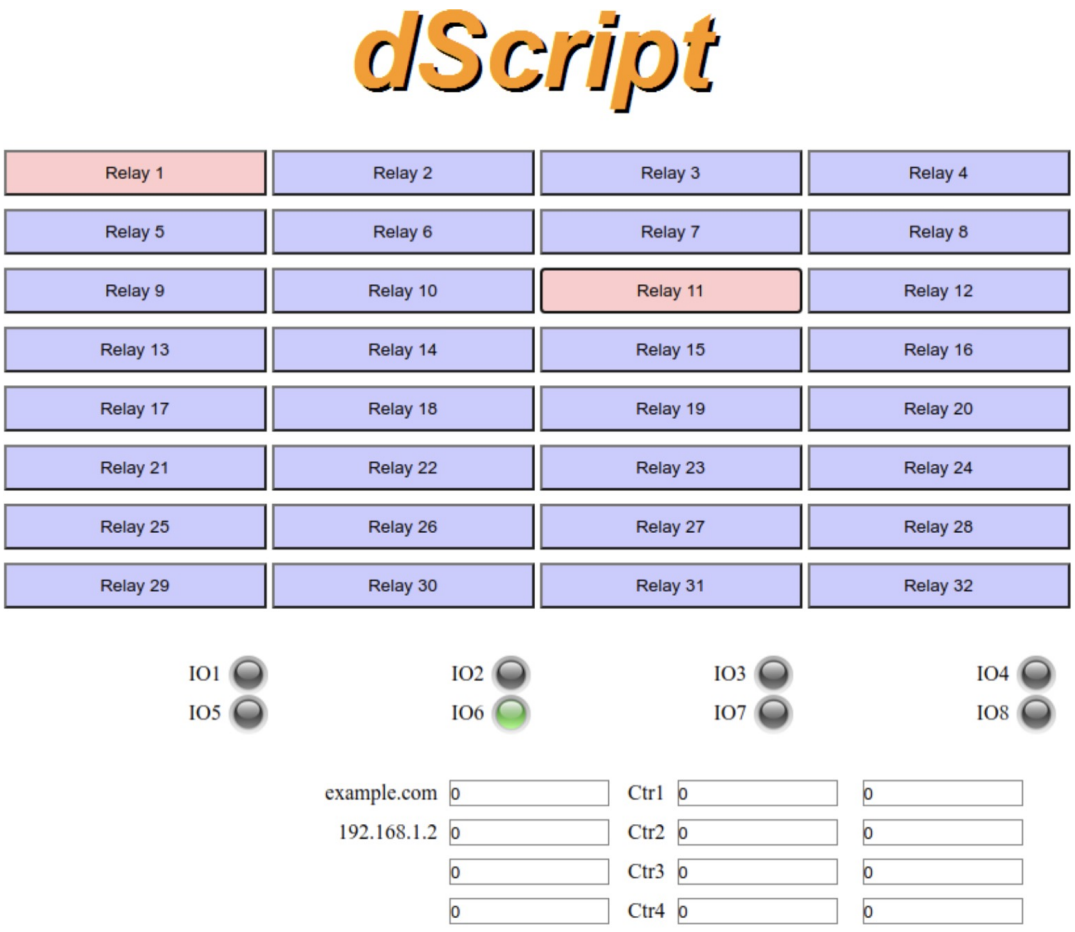
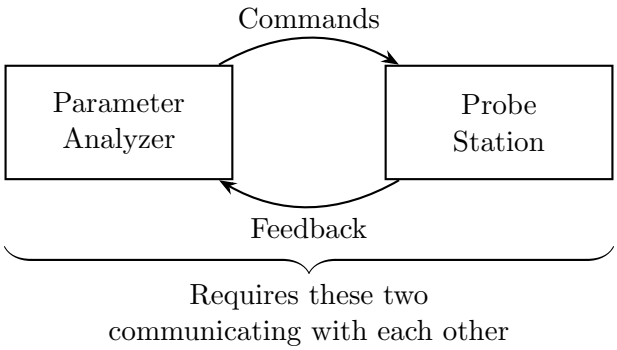


Figure 20: Screenshot of the previous manual GUI for comparison

3 The Probe Station

The project configured an MPI TS2000-D probe station and a Keithley 4200A-SCS parameter analyzer for remote control. Diagnostic assessments revealed a hardware issue with the prober's GPIB module. We resolved this by connecting an external GPIB module and reconfiguring the prober, enabling successful communication and automating the wafer test procedure.



3.1 Task Description

The MPI TS2000-D Probe Station has the capability to be remotely controlled. A Keithley 4200A-SCS Parameter Analyzer can remotely control the probe station, allowing for automated routines and scripts to map a wafer. This automated communication between the devices enables automated routines and scripts to map a wafer. This simplifies the wafer measurement process.

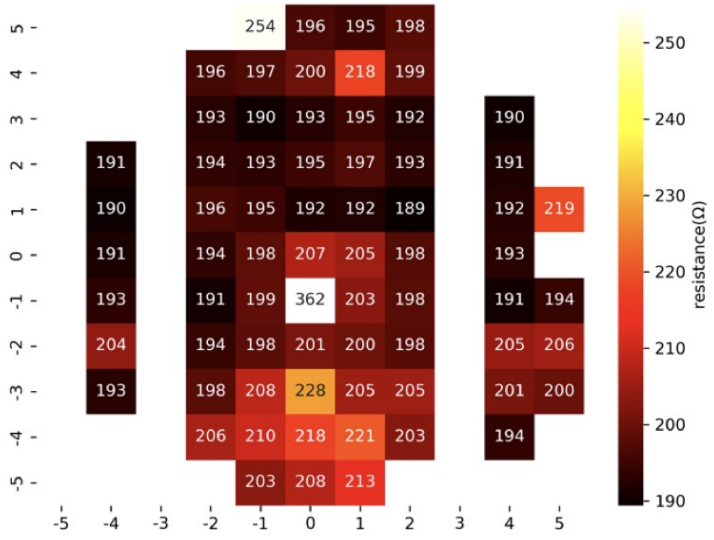


Figure 21: Resistance mapping of a sample wafer (Takes a full day for three people to complete manually)

3.2 Implementation

After some troubleshooting, we found an internal part of the MPI machine, a USB-GPIB converter, that was malfunctioning. We used an external converter and configured the software to use the new converter and therefore we were able to make these devices communicate.

3.3 Results and Discussion

The integration of the external GPIB module and the subsequent reconfiguration resolved the communication issue. This solution led to an improved automated process for wafer testing. The efficiency of the system was significantly enhanced, reducing the time and manpower required for tasks such as resistance mapping of a sample wafer.

Acknowledgments

I would like to extend my deepest gratitude to Professor Tobias J. Kippenberg for accepting me into his lab and providing support and guidance throughout my internship. His mentorship has been invaluable in my journey and has greatly contributed to the success of my projects.

In particular, I want to express my profound thanks to my supervisor, Dr. Marco Scigliuzzo for his extensive guidance on this project. His expertise, patience, and commitment have been instrumental in my development and understanding and his willingness to offer timely advice significantly enhanced the quality of my work.

I'm deeply grateful to Evgenii Guzovskii and Mahdi Chegnizadeh for their invaluable assistance; their support went above and beyond what I could have hoped for. My appreciation also goes to Amir Youssefi, Dr. Shingo Kono, and Amirali Arabmoheghi for sharing their insights and perspectives.

Finally, a huge thank you to all the members of the K-lab for their constant support and encouragement. The collaborative and inspiring atmosphere of the lab has significantly contributed to my personal and professional growth.

References

[1] Julian Kelly, Peter O'Malley, Matthew Neeley, Hartmut Neven, and John M Martinis. Physical qubit calibration on a directed acyclic graph. *arXiv preprint arXiv:1803.03226*, 2018.

[2] P. Krantz, M. Kjaergaard, F. Yan, T. P. Orlando, S. Gustavsson, and W. D. Oliver. A quantum engineer's guide to superconducting qubits. *Applied Physics Reviews*, 6(2), June 2019.